LISTING PROGRAM DATABASE

Koneksi.php

```
<?php
        $server         = "localhost"; //sesuaikan dengan nama server
        $user           = "bayu"; //sesuaikan username
        $password       = "bayu"; //sesuaikan password
        $database       = "monitoring"; //sesuaikan target databese


$con = mysqli_connect($server, $user, $password, $database);
         if (mysqli_connect_errno()) {
                echo "Gagal terhubung MySQL: " . mysqli_connect_error();
         }

?>
```

Login.php

```
<?php

        include_once "koneksi.php";

        class usr{}

         $username = $_POST["username"];
         $password = $_POST["password"];

         if ((empty($username)) || (empty($password))) {
                $response = new usr();
                $response->success = 0;
                $response->message = "Kolom tidak boleh kosong";
                die(json_encode($response));
         }

          $query = mysqli_query($con, "SELECT * FROM user WHERE
username='$username' AND password='$password'");

         $row = mysqli_fetch_array($query);

         if (!empty($row)){
                $response = new usr();
                $response->success = 1;
                $response->message = "Selamat datang ".$row['username'];
                $response->id = $row['id'];
                $response->username = $row['username'];
                die(json_encode($response));

        } else {
                $response = new usr();
                $response->success = 0;
                $response->message = "Username atau password salah";
                die(json_encode($response));
         }

         mysqli_close($con);

?>
```

LISTING PROGRAM ANDROID STUDIO

AppController.Java

```
/*
 * Copyright (c) MoCam 2019
 */

package cc.echonet.coolmicapp.app;

import android.app.Application;
import android.text.TextUtils;

import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.toolbox.Volley;

public class AppController extends Application {
public static final String TAG = AppController.class.getSimpleName();

private RequestQueue mRequestQueue;

private static AppController mInstance;

    @Override
public void onCreate() {
super.onCreate();
mInstance = this;
    }

public static synchronized AppController getInstance() {
return mInstance;
    }

public RequestQueue getRequestQueue() {
if (mRequestQueue == null) {
mRequestQueue = Volley.newRequestQueue(getApplicationContext());
        }

return mRequestQueue;
    }

public <T>void addToRequestQueue(Request<T> req, String tag) {
        req.setTag(TextUtils.isEmpty(tag) ? TAG : tag);
        getRequestQueue().add(req);
    }

public <T>void addToRequestQueue(Request<T> req) {
        req.setTag(TAG);
        getRequestQueue().add(req);
    }

public void cancelPendingRequests(Object tag) {
if (mRequestQueue != null) {
mRequestQueue.cancelAll(tag);
        }
    }
}
```

Aboutme.java

```
package cc.echonet.coolmicapp;

import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;

public class aboutme extends AppCompatActivity {
    @Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
        setContentView(R.layout.aboutme);

    }
}
```

Galeri.Java

```
package cc.echonet.coolmicapp;

import android.os.Bundle;
import android.os.Environment;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import android.widget.Toast;

import java.io.File;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;

import app.ltaps.imagegallery.ImageGallery;
import app.ltaps.imagegallery.Utils.LanguageHelper;
import app.ltaps.imagegallery.Utils.OnClickCallback;

public class Galeri extends AppCompatActivity {

    @Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
        setContentView(R.layout.galeri);

        ImageGallery imageGallery = (ImageGallery)
findViewById(R.id.imageGallery);
        imageGallery
                .setImages(getImages())
//.setImages(getImagesAsArray())
.setLanguageHelper(new LanguageHelper(this)
                        .setNoImagesAvailable("No images are available!")
                        .setOutOf("out of"))
                .setOnLargeImageClickCallback(new OnClickCallback() {
                    @Override
public void OnClick(String currentImageUrl) {
                        Toast.makeText(getApplicationContext(), "Clicked
image", Toast.LENGTH_LONG).show();
                    }
                })
                .start();
    }

private List<String> getImages() {
        List<String> imageUrls = new ArrayList<>();
```

```
        String path =
Environment.getExternalStorageDirectory().getPath()+"/Screenshot/";
        File directory = new File(path); //path is the string specifying
your directory path.
File[] files = directory.listFiles();
for (int i = 0; i < files.length; i++)
        {
            Log.d("Files", "FileName:"+path + files[i].getName()); //these
are the different filenames in the directory

//You can now use these file names along with the directory path and
convert the image there to a bitmap and set it to recycler view's image
view

File imgFile = new  File(path + files[i].getName());
if(imgFile.exists()){
                imageUrls.add(path + files[i].getName());
//                Bitmap myBitmap =
BitmapFactory.decodeFile(imgFile.getAbsolutePath()); //this is the bitmap
for the image
//
//                ImageView myImage = (ImageView)
findViewById(R.id.imageviewTest);//your image view in the recycler view
//
//                myImage.setImageBitmap(myBitmap);//image set to the image
view

};
        }
//
imageUrls.add("http://www.allgoodcleanrecords.com/photo/record_stores/route
66/DSC_0172.jpg");
//
imageUrls.add("http://www.allgoodcleanrecords.com/photo/record_stores/route
66/DSC_0177.jpg");
//
imageUrls.add("http://www.allgoodcleanrecords.com/photo/record_stores/route
66/DSC_0176.jpg");
//
imageUrls.add("http://www.allgoodcleanrecords.com/photo/record_stores/route
66/DSC_0180.jpg");
//
imageUrls.add("http://www.allgoodcleanrecords.com/photo/record_stores/route
66/DSC_0177.jpg");
//
imageUrls.add("http://www.allgoodcleanrecords.com/photo/record_stores/route
66/DSC_0176.jpg");
//
imageUrls.add("http://www.allgoodcleanrecords.com/photo/record_stores/route
66/DSC_0180.jpg");
//
imageUrls.add("http://www.allgoodcleanrecords.com/photo/record_stores/route
66/DSC_0185.jpg");
//
imageUrls.add("http://www.allgoodcleanrecords.com/photo/record_stores/route
66/DSC_0179.jpg");
//
imageUrls.add("http://www.allgoodcleanrecords.com/photo/record_stores/route
66/DSC_0179121.jpg");
return imageUrls;
    }
```

```
private String[] getImagesAsArray() {
        Object[] objectList = getImages().toArray();
return Arrays.copyOf(objectList, objectList.length, String[].class);
    }

}
```

Login.java

```
package cc.echonet.coolmicapp;

import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.graphics.drawable.AnimationDrawable;
import android.net.ConnectivityManager;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.RelativeLayout;
import android.widget.Toast;

import com.android.volley.Request;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;


import org.json.JSONException;
import org.json.JSONObject;

import java.util.HashMap;
import java.util.Map;

import cc.echonet.coolmicapp.app.AppController;

public class Login extends AppCompatActivity{
    ProgressDialog pDialog;
    Button btn_login;
    EditText txt_username, txt_password;
    Intent intent;
    RelativeLayout myLayout;
    AnimationDrawable animationDrawable;

int success;
    ConnectivityManager conMgr;

private String url = Server.URL + "login.php";

private static final String TAG = Login.class.getSimpleName();

private static final String TAG_SUCCESS = "success";
private static final String TAG_MESSAGE = "message";

public final static String TAG_USERNAME = "username";
```

```
public final static String TAG_ID = "id";

    String tag_json_obj = "json_obj_req";

    SharedPreferences sharedpreferences;
    Boolean session = false;
    String id, username;
public static final String my_shared_preferences = "my_shared_preferences";
public static final String session_status = "session_status";

    @Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
        setContentView(R.layout.login);

myLayout = (RelativeLayout) findViewById(R.id.myLayout);

//        animationDrawable = (AnimationDrawable) myLayout.getBackground();
//        animationDrawable.setEnterFadeDuration(4500);
//        animationDrawable.setExitFadeDuration(4500);
//        animationDrawable.start();

conMgr = (ConnectivityManager)
getSystemService(Context.CONNECTIVITY_SERVICE);
        {
if (conMgr.getActiveNetworkInfo() != null
&&conMgr.getActiveNetworkInfo().isAvailable()
&&conMgr.getActiveNetworkInfo().isConnected()) {
            } else {
                Toast.makeText(getApplicationContext(), "No Internet
Connection",
                        Toast.LENGTH_LONG).show();
            }
        }

btn_login = (Button) findViewById(R.id.btn_login);

txt_username = (EditText) findViewById(R.id.txt_username);
txt_password = (EditText) findViewById(R.id.txt_password);

// Cek session login jika TRUE maka langsung buka MainActivity
sharedpreferences = getSharedPreferences(my_shared_preferences,
Context.MODE_PRIVATE);
session = sharedpreferences.getBoolean(session_status, false);
id = sharedpreferences.getString(TAG_ID, null);
username = sharedpreferences.getString(TAG_USERNAME, null);

if (session) {
            Intent intent = new Intent(Login.this, MainActivity.class);
            intent.putExtra(TAG_ID, id);
            intent.putExtra(TAG_USERNAME, username);
            finish();
            startActivity(intent);
        }


btn_login.setOnClickListener(new View.OnClickListener() {

            @Override
public void onClick(View v) {
// TODO Auto-generated method stub
```

```
String username = txt_username.getText().toString();
                String password = txt_password.getText().toString();

// mengecek kolom yang kosong
if (username.trim().length() > 0 && password.trim().length() > 0) {
if (conMgr.getActiveNetworkInfo() != null
&&conMgr.getActiveNetworkInfo().isAvailable()
&&conMgr.getActiveNetworkInfo().isConnected()) {
                        checkLogin(username, password);
                    } else {
                        Toast.makeText(getApplicationContext() ,"No
Internet Connection", Toast.LENGTH_LONG).show();
                    }
                } else {
// Prompt user to enter credentials
Toast.makeText(getApplicationContext() ,"Kolom tidak boleh kosong",
Toast.LENGTH_LONG).show();
                }
            }
        });



    }

private void checkLogin(final String username, final String password) {
pDialog = new ProgressDialog(this);
pDialog.setCancelable(false);
pDialog.setMessage("Logging in ...");
        showDialog();

        StringRequest strReq = new StringRequest(Request.Method.POST, url,
new Response.Listener<String>() {

            @Override
public void onResponse(String response) {
                Log.e(TAG, "Login Response: " + response.toString());
                hideDialog();

try {
                    JSONObject jObj = new JSONObject(response);
success = jObj.getInt(TAG_SUCCESS);

// Check for error node in json
if (success == 1) {
                        String username = jObj.getString(TAG_USERNAME);
                        String id = jObj.getString(TAG_ID);

                        Log.e("Successfully Login!", jObj.toString());

                        Toast.makeText(getApplicationContext(),
jObj.getString(TAG_MESSAGE), Toast.LENGTH_LONG).show();

// menyimpan login ke session
SharedPreferences.Editor editor = sharedpreferences.edit();
                        editor.putBoolean(session_status, true);
                        editor.putString(TAG_ID, id);
                        editor.putString(TAG_USERNAME, username);
                        editor.commit();

// Memanggil main activity
```

```
Intent intent = new Intent(Login.this, Menu_utama.class);
                        intent.putExtra(TAG_ID, id);
                        intent.putExtra(TAG_USERNAME, username);
                        finish();
                        startActivity(intent);
                    } else {
                        Toast.makeText(getApplicationContext(),
                                jObj.getString(TAG_MESSAGE),
Toast.LENGTH_LONG).show();

                    }
                } catch (JSONException e) {
// JSON error
e.printStackTrace();
                }

            }
        }, new Response.ErrorListener() {

            @Override
public void onErrorResponse(VolleyError error) {
                Log.e(TAG, "Login Error: " + error.getMessage());
                Toast.makeText(getApplicationContext(),
                        error.getMessage(), Toast.LENGTH_LONG).show();

                hideDialog();

            }
        }) {

            @Override
protected Map<String, String> getParams() {
// Posting parameters to login url
Map<String, String> params = new HashMap<String, String>();
                params.put("username", username);
                params.put("password", password);

return params;
            }

        };

// Adding request to request queue
AppController.getInstance().addToRequestQueue(strReq, tag_json_obj);
    }

private void showDialog() {
if (!pDialog.isShowing())
pDialog.show();
    }

private void hideDialog() {
if (pDialog.isShowing())
pDialog.dismiss();


    }
}
```

MainActivity.java

```
package cc.echonet.coolmicapp;

import android.app.Activity;
import android.app.AlertDialog;
import android.content.ClipData;
import android.content.ClipboardManager;
import android.content.ComponentName;
import android.content.Context;
import android.content.ContextWrapper;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.ServiceConnection;
import android.content.res.Configuration;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Picture;
import android.graphics.drawable.ColorDrawable;
import android.graphics.drawable.Drawable;
import android.graphics.drawable.TransitionDrawable;
import android.net.Uri;
import android.os.AsyncTask;
import android.os.Bundle;
import android.os.Environment;
import android.os.Handler;
import android.os.IBinder;
import android.os.Message;
import android.os.Messenger;
import android.os.RemoteException;
import android.support.annotation.NonNull;
import android.util.Log;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.view.animation.AlphaAnimation;
import android.view.animation.Animation;
import android.view.animation.LinearInterpolator;
import android.webkit.WebSettings;
import android.webkit.WebView;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.ProgressBar;
import android.widget.SeekBar;
import android.widget.TextView;
import android.widget.Toast;

import java.io.BufferedInputStream;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import java.util.Locale;

import cc.echonet.coolmicdspjava.VUMeterResult;
```

```
/**
 * This activity demonstrates how to use JNI to encode and decode
ogg/vorbis audio
 */
public class MainActivity extends Activity {
    Messenger mBackgroundService = null;
    Messenger mBackgroundServiceClient = new Messenger(new
IncomingHandler(this));
boolean mBackgroundServiceBound = false;
    Constants.CONTROL_UI currentState;

    BackgroundServiceState backgroundServiceState;


    CoolMic coolmic = null;
    Button start_button;
boolean start_button_debounce_active = false;
    SeekBar gainLeft;
    SeekBar gainRight;
    Button btnCapture;
    Animation animation = new AlphaAnimation(1, 0);

    ColorDrawable transitionColorGrey = new
ColorDrawable(Color.parseColor("#66999999"));
    ColorDrawable transitionColorRed = new ColorDrawable(Color.RED);
    ColorDrawable[] transitionColorDefault = {transitionColorGrey,
transitionColorRed};

    TransitionDrawable transitionButton = new
TransitionDrawable(transitionColorDefault);

    Drawable buttonColor;
    ImageView imageView1;
    ClipboardManager myClipboard;


private ServiceConnection mBackgroundServiceConnection = new
ServiceConnection() {
public void onServiceConnected(ComponentName className, IBinder service) {
// This is called when the connection with the service has been
            // established, giving us the object we can use to
            // interact with the service.  We are communicating with the
            // service using a Messenger, so here we get a client-side
            // representation of that from the raw IBinder object.
mBackgroundService = new Messenger(service);
mBackgroundServiceBound = true;

// Create and send a message to the service, using a supported 'what' value
Message msg = Message.obtain(null, Constants.C2S_MSG_STATE, 0, 0);

            msg.replyTo = mBackgroundServiceClient;

try {
mBackgroundService.send(msg);
                sendStreamReload();
            } catch (RemoteException e) {
                e.printStackTrace();
            }
        }

public void onServiceDisconnected(ComponentName className) {
```

```
// This is called when the connection with the service has been
            // unexpectedly disconnected -- that is, its process crashed.
mBackgroundService = null;
mBackgroundServiceBound = false;
        }
    };


/**
     * Handler of incoming messages from clients.
     */
static class IncomingHandler extends Handler {
private final MainActivity activity;

        IncomingHandler(MainActivity activity) {
this.activity = activity;
        }

        @Override
public void handleMessage(Message msg) {
            Bundle bundle = msg.getData();

switch (msg.what) {
case Constants.S2C_MSG_STATE_REPLY:
activity.backgroundServiceState = (BackgroundServiceState)
bundle.getSerializable("state");

if (activity.backgroundServiceState == null) {
return;
                    }

                    Log.v("IH", "In Handler: S2C_MSG_STATE_REPLY: State=" +
activity.backgroundServiceState.uiState.toString());

activity.controlRecordingUI(activity.backgroundServiceState.uiState);

                    ((TextView)
activity.findViewById(R.id.timerValue)).setText(activity.backgroundServiceS
tate.timerString);
                    ((TextView)
activity.findViewById(R.id.txtState)).setText(activity.backgroundServiceSta
te.txtState);
                    ((TextView)
activity.findViewById(R.id.txtListeners)).setText(activity.backgroundServic
eState.listenersString);

if
(activity.backgroundServiceState.uiState.equals(Constants.CONTROL_UI.CONTRO
L_UI_CONNECTING)) {
                        Log.v("IH", "In Handler: S2C_MSG_STATE_REPLY: X!");
                    }

break;

case Constants.S2C_MSG_ERROR:
                    String error = bundle.getString("error");

                    Log.v("IH", "In Handler: S2C_MSG_ERROR: State=" +
activity.backgroundServiceState.uiState.toString());
                    Log.v("IH", "In Handler: S2C_MSG_ERROR: error=" +
error);
```

```
                    Toast.makeText(activity, error,
Toast.LENGTH_LONG).show();

if
(activity.backgroundServiceState.uiState.equals(Constants.CONTROL_UI.CONTRO
L_UI_CONNECTING)) {
                        Log.v("IH", "In Handler: S2C_MSG_ERROR: X!");
                    }

break;
case Constants.S2C_MSG_STREAM_START_REPLY:
activity.controlVuMeterUI(Integer.parseInt(activity.coolmic.getVuMeterInter
val()) != 0);

                    Log.v("IH", "In Handler: S2C_MSG_STREAM_START_REPLY:
X!");
activity.start_button.setClickable(true);
break;

case Constants.S2C_MSG_STREAM_STOP_REPLY:
boolean was_running = bundle.getBoolean("was_running");

                    Log.v("IH", "In Handler: S2C_MSG_STREAM_STOP_REPLY:
X!");

if (was_running) {
                        Toast.makeText(activity,
R.string.broadcast_stop_message, Toast.LENGTH_SHORT).show();
                    }

activity.start_button.setClickable(true);
break;

case Constants.S2C_MSG_PERMISSIONS_MISSING:
                    Utils.requestPermissions(activity);

break;

case Constants.S2C_MSG_CONNECTION_UNSET:
                    AlertDialog.Builder alertDialog =
Utils.buildAlertDialogCMTSTOS(activity);

alertDialog.setPositiveButton(R.string.mainactivity_missing_connection_deta
ils_yes, new DialogInterface.OnClickListener() {
public void onClick(DialogInterface dialog, int which) {
                            Utils.loadCMTSData(activity, "default");
activity.startRecording(null);
                        }
                    });

                    alertDialog.show();

break;

case Constants.S2C_MSG_CMTS_TOS:
                    AlertDialog.Builder alertDialogCMTSTOS = new
AlertDialog.Builder(activity);

alertDialogCMTSTOS.setTitle(R.string.coolmic_tos_title);
                    alertDialogCMTSTOS.setMessage(R.string.coolmic_tos);
```

```
alertDialogCMTSTOS.setNegativeButton(R.string.coolmic_tos_cancel, new
DialogInterface.OnClickListener() {
public void onClick(DialogInterface dialog, int which) {
                            dialog.cancel();
                        }
                    });

alertDialogCMTSTOS.setPositiveButton(R.string.coolmic_tos_accept, new
DialogInterface.OnClickListener() {
public void onClick(DialogInterface dialog, int which) {
                            activity.startRecording(null, true);
                        }
                    });

                    alertDialogCMTSTOS.show();

break;
case Constants.S2C_MSG_VUMETER:
                    Bundle bundleVUMeter = msg.getData();

                    activity.handleVUMeterResult((VUMeterResult)
bundleVUMeter.getSerializable("vumeterResult"));

break;
case Constants.C2S_MSG_GAIN:
                    Bundle bundleGain = msg.getData();

                    activity.setGain(bundleGain.getInt("left"),
bundleGain.getInt("right"));

break;
default:
super.handleMessage(msg);
            }
        }
    }

private void setGain(int left, int right) {
        gainLeft.setProgress(left);
        gainRight.setProgress(right);
    }

private void sendGain(int left, int right) {
if (mBackgroundServiceBound) {
            Message msgReply = Message.obtain(null, Constants.C2S_MSG_GAIN,
0, 0);

            msgReply.replyTo = mBackgroundServiceClient;

            Bundle bundle = msgReply.getData();

            bundle.putInt("left", left);
            bundle.putInt("right", right);

try {
                mBackgroundService.send(msgReply);
            } catch (RemoteException e) {
                e.printStackTrace();
            }
        }
```

```
        coolmic.setVolumeLeft(left);
        coolmic.setVolumeRight(right);
    }

    @Override
public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.main_activity_menu, menu);
return true;
    }

    @Override
public boolean onOptionsItemSelected(MenuItem item) {
switch (item.getItemId()) {
case R.id.menu_action_share:
                performShare();
return true;
case R.id.menu_action_settings:
                goSettings();
return true;
case R.id.menu_action_about:
                goAbout();
return true;
case R.id.menu_action_help:
                Intent helpIntent = new Intent(Intent.ACTION_VIEW,
Uri.parse(getString(R.string.coolmic_help_url)));
                startActivity(helpIntent);
return true;
case R.id.menu_action_quit:
                exitApp();
return true;
default:
                Toast.makeText(getApplicationContext(),
R.string.menu_action_default, Toast.LENGTH_SHORT).show();
break;
        }
return true;
    }

private void exitApp() {
        stopRecording();

        disconnectService();

        stopService(new Intent(this, BackgroundService.class));

new Handler().postDelayed(new Runnable() {
            @Override
public void run() {
                stopService(new Intent(getApplicationContext(),
BackgroundService.class));
            }
        }, 250);

new Handler().postDelayed(new Runnable() {
            @Override
public void run() {
                System.exit(0);
            }
        }, 500);
```

```
    }

private void goSettings() {
        Intent i = new Intent(MainActivity.this, SettingsActivity.class);
        startActivity(i);
    }

private void goAbout() {
        Intent i = new Intent(MainActivity.this, AboutActivity.class);
        startActivity(i);
    }

    @Override
public void onConfigurationChanged(Configuration newConfig) {
super.onConfigurationChanged(newConfig);
if (newConfig.orientation == Configuration.ORIENTATION_LANDSCAPE) {
            imageView1.getLayoutParams().height = 60;
        } else {
            imageView1.getLayoutParams().height = 400;
        }
    }

    @Override
public void onDestroy() {
super.onDestroy();

        Log.v("$$$$$$", "In Method: onDestroy()");

this.exitApp();
    }

private void connectService() {
if (!mBackgroundServiceBound) {
            Intent intent = new Intent(this, BackgroundService.class);
            startService(intent);
            bindService(intent, mBackgroundServiceConnection,
Context.BIND_AUTO_CREATE);
        }
    }

private void disconnectService() {
if (mBackgroundServiceBound) {
            unbindService(mBackgroundServiceConnection);
            mBackgroundServiceBound = false;
        }
    }

    @Override
protected void onStart() {
super.onStart();
// Bind to the service
connectService();
        controlRecordingUI(currentState);
    }

    @Override
protected void onStop() {
super.onStop();
// Unbind from the service
disconnectService();
```

```
    }

    Button btn_exit,btn_capture;
private WebView webView = null;
private AsyncTask mMyTask;

    @Override
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);

        setContentView(R.layout.home);

        imageView1 = (ImageView) findViewById(R.id.imageView1);
this.webView = (WebView) findViewById(R.id.webview);
        btn_exit = (Button) findViewById(R.id.btn_exit);
        btn_capture = (Button) findViewById(R.id.btn_capture);
        btn_exit.setOnClickListener(new View.OnClickListener() {
            @Override
public void onClick(View v) {
                Intent intent = new Intent(MainActivity.this,
Menu_utama.class);
//                finish();
startActivity(intent);
            }
        });
        btn_capture.setOnClickListener(new View.OnClickListener() {
            @Override
public void onClick(View view) {
               getSnapshot();
            }
        });


        WebSettings webSettings = webView.getSettings();
        webSettings.setJavaScriptEnabled(true);

        streamingclient webViewClient = new streamingclient(this);
        webView.setWebViewClient(webViewClient);

        webView.loadUrl("http://193.168.195.248:8090/?action=stream");
//getBitmapFromURL("http://193.168.195.248:8090/?action=stream");

Log.v("onCreate", (imageView1 == null ? "iv null" : "iv ok"));

if (imageView1 != null) {
            imageView1.setOnClickListener(new View.OnClickListener() {
                @Override
public void onClick(View v) {
                    onImageClick(v);
                }
            });
        }

        myClipboard = (ClipboardManager)
getSystemService(CLIPBOARD_SERVICE);

        animation.setDuration(500); // duration - half a second
animation.setInterpolator(new LinearInterpolator()); // do not alter
animation rate
animation.setRepeatCount(Animation.INFINITE); // Repeat animation
infinitely
```

```
animation.setRepeatMode(Animation.REVERSE);
        start_button = (Button) findViewById(R.id.start_recording_button);

        gainLeft = (SeekBar) findViewById(R.id.pbGainMeterLeft);
        gainRight = (SeekBar) findViewById(R.id.pbGainMeterRight);

        SeekBar.OnSeekBarChangeListener seekBarChangeListener = new
SeekBar.OnSeekBarChangeListener() {

            @Override
public void onProgressChanged(SeekBar seekBar, int i, boolean b) {
if (b) {
if (backgroundServiceState.channels != 2) {
int value = seekBar.getProgress();

                        setGain(value, value);
                    }

                    MainActivity.this.sendGain(gainLeft.getProgress(),
gainRight.getProgress());
                }
            }

            @Override
public void onStartTrackingTouch(SeekBar seekBar) {

            }

            @Override
public void onStopTrackingTouch(SeekBar seekBar) {

            }
        };

        gainLeft.setOnSeekBarChangeListener(seekBarChangeListener);
        gainRight.setOnSeekBarChangeListener(seekBarChangeListener);

        buttonColor = start_button.getBackground();

        coolmic = new CoolMic(this, "default");

        controlVuMeterUI(Integer.parseInt(coolmic.getVuMeterInterval()) !=
0);

        controlRecordingUI(currentState);

        setGain(coolmic.getVolumeLeft(), coolmic.getVolumeRight());

        start_button.setOnLongClickListener(new View.OnLongClickListener()
{
            @Override
public boolean onLongClick(View view) {
                sendStreamReload();
return true;
            }
        });

        start_button.setOnClickListener(new View.OnClickListener() {
            @Override
public void onClick(View v) {
if (start_button_debounce_active)
```

```
return;
                start_button_debounce_active = true;
                v.postDelayed(new Runnable() {
                    @Override
public void run() {
                        start_button_debounce_active = false;
                    }
                }, 500);
                start_button.setClickable(false);
                startRecording(v);
            }
        });
    }

public static Bitmap getBitmapFromURL(String src) {
try {
            URL url = new URL(src);
            HttpURLConnection connection = (HttpURLConnection)
url.openConnection();
            connection.setDoInput(true);
            connection.connect();
            InputStream input = connection.getInputStream();
            Bitmap myBitmap = BitmapFactory.decodeStream(input);
return myBitmap;
        } catch (IOException e) {
// Log exception
return null;
        }
    }

private void sendStreamReload() {
if (mBackgroundServiceBound) {
            System.out.println("MainActivity.sendStreamReload: We have a
background service!");
            Message msgReply = Message.obtain(null,
Constants.C2S_MSG_STREAM_RELOAD, 0, 0);

            msgReply.replyTo = mBackgroundServiceClient;

try {
                mBackgroundService.send(msgReply);
            } catch (RemoteException e) {
                e.printStackTrace();
            }
        } else {
            System.out.println("MainActivity.sendStreamReload: No
background service!");
        }
    }

public void onImageClick(View view) {
if (coolmic.isConnectionSet()) {
            ClipData myClip = ClipData.newPlainText("text",
coolmic.getStreamURL());
            myClipboard.setPrimaryClip(myClip);
            Toast.makeText(getApplicationContext(),
R.string.mainactivity_broadcast_url_copied, Toast.LENGTH_SHORT).show();
        } else {
            Toast.makeText(getApplicationContext(),
R.string.mainactivity_connectiondetails_unset, Toast.LENGTH_SHORT).show();
        }
```

```
    }

public void performShare() {
if (coolmic.isConnectionSet()) {
            Intent shareIntent = new Intent();
            shareIntent.setAction(Intent.ACTION_SEND);
            shareIntent.putExtra(Intent.EXTRA_TEXT,
coolmic.getStreamURL());
            shareIntent.setType("text/plain");

            startActivity(Intent.createChooser(shareIntent,
getResources().getText(R.string.menu_action_share_title)));

        } else {
            Toast.makeText(getApplicationContext(),
R.string.mainactivity_connectiondetails_unset, Toast.LENGTH_SHORT).show();
        }
    }

    @Override
public void onBackPressed() {
    }

private long getChannels() {
if (currentState == Constants.CONTROL_UI.CONTROL_UI_CONNECTED) {
return backgroundServiceState.channels;
        }

if (coolmic != null) {
return Integer.parseInt(coolmic.getChannels());
        }

return 2; // default.
}

public void controlRecordingUI(Constants.CONTROL_UI state) {
        View sb;

        sb = findViewById(R.id.pbGainMeterLeft);
if (this.getChannels() == 2) {
            sb.setVisibility(View.VISIBLE);
        } else {
            sb.setVisibility(View.GONE);
        }

if (state == currentState) {
return;
        }

if (state == Constants.CONTROL_UI.CONTROL_UI_CONNECTING) {
            start_button.startAnimation(animation);
            start_button.setBackground(transitionButton);
            transitionButton.startTransition(5000);

            start_button.setText(R.string.cmdStartInitializing);
        } else if (state == Constants.CONTROL_UI.CONTROL_UI_CONNECTED) {
            start_button.startAnimation(animation);
            start_button.setBackground(transitionButton);
            transitionButton.startTransition(5000);

            start_button.setText(R.string.broadcasting);
```

```
        } else {
            start_button.clearAnimation();
            start_button.setBackground(buttonColor);
            start_button.setText(R.string.start_broadcast);

            ((ProgressBar)
MainActivity.this.findViewById(R.id.pbVuMeterLeft)).setProgress(0);
            ((ProgressBar)
MainActivity.this.findViewById(R.id.pbVuMeterRight)).setProgress(0);
            ((TextProgressBar)
MainActivity.this.findViewById(R.id.pbVuMeterLeft)).setText("");
            ((TextProgressBar)
MainActivity.this.findViewById(R.id.pbVuMeterRight)).setText("");
            ((TextView)
MainActivity.this.findViewById(R.id.rbPeakLeft)).setText("");
            ((TextView)
MainActivity.this.findViewById(R.id.rbPeakRight)).setText("");
        }

        currentState = state;
    }

public void controlVuMeterUI(boolean visible) {
if (findViewById(R.id.llVuMeterLeft) != null) {
            findViewById(R.id.llVuMeterLeft).setVisibility((visible ?
View.VISIBLE : View.GONE));
        }

if (findViewById(R.id.llVuMeterRight) != null) {
            findViewById(R.id.llVuMeterRight).setVisibility((visible ?
View.VISIBLE : View.GONE));
        }

        findViewById(R.id.pbVuMeterLeft).setVisibility((visible ?
View.VISIBLE : View.GONE));
        findViewById(R.id.pbVuMeterRight).setVisibility((visible ?
View.VISIBLE : View.GONE));
        findViewById(R.id.rbPeakLeft).setVisibility((visible ? View.VISIBLE
: View.GONE));
        findViewById(R.id.rbPeakRight).setVisibility((visible ?
View.VISIBLE : View.GONE));
    }

public void startRecording(View view) {
        startRecording(view, true);
    }

public void startRecording(View view, boolean cmtsTOSAccepted) {
if (mBackgroundServiceBound) {
            Message msgReply = Message.obtain(null,
Constants.C2S_MSG_STREAM_ACTION, 0, 0);

            msgReply.replyTo = mBackgroundServiceClient;

            Bundle bundle = msgReply.getData();

            bundle.putString("profile", "default");
            bundle.putBoolean("cmtsTOSAccepted", cmtsTOSAccepted);

try {
mBackgroundService.send(msgReply);
```

```
            } catch (RemoteException e) {
                e.printStackTrace();
            }
        }
    }

public void stopRecording() {
if (mBackgroundServiceBound) {
            Message msgReply = Message.obtain(null,
Constants.C2S_MSG_STREAM_STOP, 0, 0);

            msgReply.replyTo = mBackgroundServiceClient;

try {
mBackgroundService.send(msgReply);
            } catch (RemoteException e) {
                e.printStackTrace();
            }
        }
    }

    @Override
public void onRequestPermissionsResult(int requestCode, @NonNull String[]
permissions, @NonNull int[] grantResults) {
if (!Utils.onRequestPermissionsResult(this, requestCode, permissions,
grantResults)) {
super.onRequestPermissionsResult(requestCode, permissions, grantResults);
        } else {
            startRecording(null);
        }
    }


static int normalizeVUMeterPower(double power) {
int g_p = (int) ((60. + power) * (100. / 60.));

if (g_p > 100) {
            g_p = 100;
        }

if (g_p < 0) {
            g_p = 0;
        }

return g_p;
    }

static String normalizeVUMeterPeak(int peak) {
if (peak == -32768 || peak == 32767) {
return "P";
        } else if (peak < -30000 || peak > 30000) {
return "p";
        } else if (peak < -8000 || peak > 8000) {
return "g";
        } else {
return "";
        }
    }

static String normalizeVUMeterPowerString(double power) {
if (power < -100) {
```

```
return "-100";
        } else if (power > 0) {
return "0";
        } else {
return String.format(Locale.ENGLISH, "%.2f", power);
        }
    }

public void handleVUMeterResult(VUMeterResult vuMeterResult) {
        TextProgressBar pbVuMeterLeft = (TextProgressBar)
this.findViewById(R.id.pbVuMeterLeft);
        TextProgressBar pbVuMeterRight = (TextProgressBar)
this.findViewById(R.id.pbVuMeterRight);

        TextView rbPeakLeft = (TextView)
this.findViewById(R.id.rbPeakLeft);
        TextView rbPeakRight = (TextView)
this.findViewById(R.id.rbPeakRight);

if (vuMeterResult.channels < 2) {

pbVuMeterLeft.setProgress(normalizeVUMeterPower(vuMeterResult.global_power)
);
            pbVuMeterLeft.setTextColor(vuMeterResult.global_power_color);

pbVuMeterLeft.setText(normalizeVUMeterPowerString(vuMeterResult.global_powe
r));

pbVuMeterRight.setProgress(normalizeVUMeterPower(vuMeterResult.global_power
));
            pbVuMeterRight.setTextColor(vuMeterResult.global_power_color);

pbVuMeterRight.setText(normalizeVUMeterPowerString(vuMeterResult.global_pow
er));

rbPeakLeft.setText(normalizeVUMeterPeak(vuMeterResult.global_peak));
            rbPeakLeft.setTextColor(vuMeterResult.global_peak_color);

rbPeakRight.setText(normalizeVUMeterPeak(vuMeterResult.global_peak));
            rbPeakRight.setTextColor(vuMeterResult.global_peak_color);
        } else {

pbVuMeterLeft.setProgress(normalizeVUMeterPower(vuMeterResult.channels_powe
r[0]));

pbVuMeterLeft.setTextColor(vuMeterResult.channels_power_color[0]);

pbVuMeterLeft.setText(normalizeVUMeterPowerString(vuMeterResult.channels_po
wer[0]));

pbVuMeterRight.setProgress(normalizeVUMeterPower(vuMeterResult.channels_pow
er[1]));

pbVuMeterRight.setTextColor(vuMeterResult.channels_power_color[1]);

pbVuMeterRight.setText(normalizeVUMeterPowerString(vuMeterResult.channels_p
ower[1]));

rbPeakLeft.setText(normalizeVUMeterPeak(vuMeterResult.channels_peak[0]));
            rbPeakLeft.setTextColor(vuMeterResult.channels_peak_color[0]);
```

```
rbPeakRight.setText(normalizeVUMeterPeak(vuMeterResult.channels_peak[1]));
            rbPeakRight.setTextColor(vuMeterResult.channels_peak_color[1]);
        }
    }
private void getSnapshot() {
        Picture picture = webView.capturePicture();
int width = picture.getWidth();
int height = picture.getHeight();
if (width > 0 && height > 0) {
            Bitmap bitmap = Bitmap.createBitmap(width, height,
Bitmap.Config.ARGB_8888);
            Canvas canvas = new Canvas(bitmap);
            picture.draw(canvas);
try {
                String fileName =
Environment.getExternalStorageDirectory().getPath()+"/Screenshot/"+System.n
anoTime()+"_webview_capture1.jpg";
                FileOutputStream fos = new FileOutputStream(fileName);
//压缩bitmap到输出流中
bitmap.compress(Bitmap.CompressFormat.JPEG, 70, fos);
                fos.close();
                Toast.makeText(MainActivity.this, "Success",
Toast.LENGTH_LONG).show();
                bitmap.recycle();
            } catch (Exception e) {
                Log.e("Error", e.getMessage());
            }
        }
    }

}
```

menu_utama.java

```
package cc.echonet.coolmicapp;

import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class Menu_utama extends AppCompatActivity{
    Button btn_logout, btn_live, btn_galeri, btn_aboutme;
    SharedPreferences sharedpreferences;
    @Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
        setContentView(R.layout.menu);


btn_logout = (Button) findViewById(R.id.btn_logout);
btn_live = (Button) findViewById(R.id.btn_live);
btn_galeri = (Button) findViewById(R.id.btn_galeri);
btn_aboutme = (Button) findViewById(R.id.btn_aboutme);
sharedpreferences = getSharedPreferences(Login.my_shared_preferences,
Context.MODE_PRIVATE);

btn_live.setOnClickListener(new View.OnClickListener() {
```

```
            @Override
public void onClick(View v) {
                Intent intent = new Intent(Menu_utama.this,
MainActivity.class);
                finish();
                startActivity(intent);

            }
        });
btn_galeri.setOnClickListener(new View.OnClickListener() {
            @Override
public void onClick(View v) {
                Intent intent = new Intent(Menu_utama.this, Galeri.class);

                startActivity(intent);
            }});
btn_aboutme.setOnClickListener(new View.OnClickListener() {
            @Override
public void onClick(View view) {
                Intent intent = new Intent(Menu_utama.this, aboutme.class);
                startActivity(intent);
            }
        });
btn_logout.setOnClickListener(new View.OnClickListener() {
            @Override
public void onClick(View v) {
// TODO Auto-generated method stub
// update login session ke FALSE dan mengosongkan nilai id dan username
SharedPreferences.Editor editor = sharedpreferences.edit();
                editor.putBoolean(Login.session_status, false);
                editor.commit();
                Intent intent = new Intent(Menu_utama.this, Login.class);
                startActivity(intent);
                finish();
            }
        });


    }
}
```

Server.java

```
/*
 * Copyright (c) MoCam 2019
 */

package cc.echonet.coolmicapp;

public class Server {
public static final String URL = "http://latekkom2019.com/bayu/Raspi/";
}
```

Streamingclient.java

```
package cc.echonet.coolmicapp;

import android.app.Activity;
import android.content.Intent;
import android.net.Uri;
import android.webkit.WebView;
```

```
import android.webkit.WebViewClient;


public class streamingclient extends WebViewClient {

private Activity activity = null;

public streamingclient(Activity activity) {
this.activity = activity;
    }

    @Override
public boolean shouldOverrideUrlLoading(WebView webView, String url) {
if(url.indexOf("journaldev.com") > -1 ) return false;

        Intent intent = new Intent(Intent.ACTION_VIEW, Uri.parse(url));
activity.startActivity(intent);
return true;
    }

}
```

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="cc.echonet.coolmicapp">

<uses-permission android:name="android.permission.RECORD_AUDIO" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>

<uses-permission android:name="android.permission.STORAGE" />

<application
        android:name=".app.AppController"
        android:icon="@drawable/icon"
        android:theme="@style/Theme.AppCompat"
        android:label="@string/app_name">
<activity
            android:name=".MainActivity"
            android:configChanges="keyboardHidden|orientation|screenSize"
            android:label="@string/app_name"
            android:launchMode="singleTop"
            android:theme="@style/AppTheme"
            android:screenOrientation="portrait">

</activity>

<activity
            android:name=".SettingsActivity"
            android:label="@string/title_activity_settings"
            android:parentActivityName=".MainActivity">
<meta-data
                android:name="android.support.PARENT_ACTIVITY"
                android:value="cc.echonet.coolmicapp.MainActivity" />
</activity>
<activity
            android:name=".AboutActivity"
```

```
            android:parentActivityName=".MainActivity">
<meta-data
                android:name="android.support.PARENT_ACTIVITY"
                android:value="cc.echonet.coolmicapp.MainActivity" />
</activity>


<activity android:name=".Login">
<intent-filter>
<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>

</activity>
<activity android:name=".Menu_utama"/>
<activity android:name=".Galeri" />
<activity android:name=".aboutme" />

<service
            android:name=".BackgroundService"
            android:enabled="true"
            android:exported="false" />
</application>

</manifest>
```